MCCARTHY & ASSOCIATES
500 WEST MAIN STE. 609
OKLAHOMA CITY, OKLAHOMA 73102
TELEPHONE: (405) 601-2798
FAX: (405) 601-2853

FACSIMILE TRANSMISSION COVER SHEET

DATE: February 25, 2008

NUMBER OF PAGES INCLUDING THIS COVER SHEET:

FROM: Mitchell K. McCarthy, Registration No. 38,794

TO: Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

ADDRESSEE/ORGANIZATION	FAX NUMBER	TELEPHONE NUMBER
Art Group 2189	(571) 273-8300	(571) 272-4100

RE: **Application No.** **10/658,982**
In re application of: **Clark E. Lubbers**
Assignee: **SEAGATE TECHNOLOGY LLC**
Dkt. No.: **STL11421**

IF YOU DO NOT RECEIVE ALL OF THE PAGES OR IF ANY ARE ILLEGIBLE, PLEASE CONTACT US AT (405) 601-2798 AS SOON AS POSSIBLE.

****

CONFIDENTIALITY NOTICE

Transmitted by Facsimile on February 25, 2008

PATENT
Docket No. STL11421

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Clark E. Lubbers, Randy L. Roberson**

Application No.: **10/658,982**
Filed: **09/10/2003**
For: **ADAPTIVE MAPPING**

Group No.: **2189**
Examiner: **Horace L. Flournay**

Mail Stop Appeal Briefs – Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF
(PATENT APPLICATION–37 C.F.R. § 41.37)

1. Transmitted herewith, is the APPEAL BRIEF in this application, with respect to the Notice of Appeal filed on August 3, 2007.

2. STATUS OF APPLICANT

This application is on behalf of other than a small entity.

3. FEE FOR FILING APPEAL BRIEF

Pursuant to 37 C.F.R. § 41.20(b)(2), the fee for filing the Appeal Brief is:

other than a small entity	$510.00
Appeal Brief fee due	**$510.00**

4. EXTENSION OF TERM

The proceedings herein are for a patent application and the provisions of 37 C.F.R. § 1.136 apply.

Applicant believes that no extension of term is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

5. TOTAL FEE DUE

The total fee due is:

Appeal brief fee	$510.00
Extension fee (if any)	$0.00
TOTAL FEE DUE	**$510.00**

6. FEE PAYMENT

Payment in the amount of $510.00 is being made with a Credit Card Authorization form (PTO-2038) and is enclosed herewith.

7. FEE DEFICIENCY

If any additional extension and/or fee is required, and if any additional fee for claims is required, charge Deposit Account No. 50-4124.

Date: 2/25/2008

Mitchell K. McCarthy
Registration No. 38,794
McCarthy & Associates
500 West Main Street, Ste. 609
OKC, OK 73102
405-601-2798
www.mccarthyiplaw.com
Customer No. 71222

Transmitted by Facsimile on February 25, 2008

PATENT
Dkt. STL11421

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: **Clark E. Lubbers and Randy L. Roberson**
Assignee: **SEAGATE TECHNOLOGY LLC**
Application No.: **10/658,982**
Filed: **September 10, 2003**
For: **ADAPTIVE MAPPING**

Group Art Unit: **2189**
Examiner: **Horace L. Flournoy**

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P. O. Box 1450
Alexandria, Virginia 22313-1450

ATTENTION: Board of Patent Appeals and Interferences

APPELLANT'S BRIEF

Appellant's Brief is in furtherance of the Notice of Appeal filed August 3, 2007. This brief contains these items under the following headings, and in the order set forth below:

I. REAL PARTY IN INTEREST

The real party in interest in this Appeal is Seagate Technology LLC.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in this Appeal.

III. STATUS OF CLAIMS

The status of the claims in this application is:

Claim	Status
1. (Original)	Independent.
2. (Original)	Depends from claim 1.
3. (Original)	Depends from claim 1.
4. (Original)	Depends from claim 3.
5. (Original)	Depends from claim 3.
6. (Canceled)	
7. (Original)	Depends from claim 3.
8. (Original)	Depends from claim 3.
9. (Original)	Depends from claim 3.
10. (Canceled)	
11. (Canceled)	
12. (Canceled)	
13. (Canceled)	
14. (Canceled)	
15. (Canceled)	
16. (Original)	Independent.
17. (Original)	Depends from claim 16.
18. (Original)	Depends from claim 17.
19. (Original)	Depends from claim 17.
20. (Canceled)	
21. (Original)	Depends from claim 17.
22. (Original)	Depends from claim 17.
23. (Original)	Depends from claim 17.
24. (Canceled)	
25. (Original)	Depends from claim 17.
26. (Previously presented)	Depends from claim 1.
27. (Previously presented)	Depends from claim 26.
28. (Previously presented)	Depends from claim 26.
29. (Previously presented)	Independent.
30. (Previously presented)	Depends from claim 29.
31. (Previously presented)	Depends from claim 30.
32. (Previously presented)	Depends from claim 29.
33. (Previously presented)	Depends from claim 29.

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application: 1-5, 7-9, 16-19, 21-23, 25-33.

B. STATUS OF ALL THE CLAIMS

1. Claims canceled: 6, 10-15, 20, and 24.
2. Claims withdrawn from consideration but not canceled: none
3. Claims pending: 1-5, 7-9, 16-19, 21-23, 25-33.
4. Claims allowed: none
5. Claims rejected: 1-5, 7-9, 16-19, 21-23, 25-33.
6. Claims objected to: none.

C. CLAIMS ON APPEAL

Claims now on appeal: 1-5, 7-9, 16-19, 21-23, 25-33.

IV. STATUS OF AMENDMENTS

Appellant filed an after-final amendment on May 29, 2007 that proposed additional language to independent claims 1, 16, and 29 to more particularly point out and distinctly claim the structural differences between the two featured directory structures, the *sparse directory structure* and the *fully populated directory structure*. The Office did not enter that amendment, stating that the proposed language raised new issues requiring additional searching.[1]

V. SUMMARY OF CLAIMED SUBJECT MATTER

Claim 1 features a method (such as 200 depicted in FIG. 2) of maintaining a directory for a data container (such as 104, see pg. 5 lines 3-4, pg. 8 lines 32-33). Steps in the method include determining whether a sparse directory structure is to be changed (such as in block 210 of FIG. 2, see pg. 6 lines 1-2). The method also includes the step of reconstructing the sparse directory structure into a fully populated directory structure (such as in block 212 of

---

[1] Advisory Action of 6/22/2007

FIG. 2, see pg. 5 lines 8-10). Alternatively, the method can include the step of reconstructing the fully populated directory structure into a sparsely populated directory structure (see pg. 4 line 31 to pg. 5 line 2, pg. 6 lines 21-27).

The method can be characterized by a sparse directory structure (such as 300 depicted in FIG. 3) defined by a plurality of first directory entries (such as 310 in FIG. 3, see pg. 7 lines 3-6). The first directory entries can include an address to one of respective addressable spaces in the storage container (such as "VALUE" in FIG. 4, see pg. 7 lines 29-31). The first directory entries can also include a descriptor (such as "INDEX" in FIG. 4, see pg. 7 lines 27-28). The first directory entries can also include a link that points to a different one of the first directory entries (such as "FORWARD LINK" and "BACKWARD LINK" in FIG. 4, see pg. 7 lines 24-25, pg. 7 line 32 to pg. 8 line 5).

The sparse directory structure can be further characterized by a bottom level list of a plurality of the first directory entries (such as 306, 308 depicted in FIG. 3, see pg. 6 lines 29-30). The sparse directory structure can be further characterized by a top level entry for each of the bottom level lists (such as 304 depicted in FIG. 3, see pg. 6 lines 29-30, pg. 7 lines 8-9). The sparse directory structure can further be characterized by a top level list that includes a plurality of the top level entries (such as 302 depicted in FIG. 3, see pg. 6 lines 29-30). In some embodiments the top level list can be a skip list, a linked list, or an ordered array (see pg. 7 lines 12-16). In some embodiments the bottom level lists can be a skip list or a linked list (see pg. 6 lines 31-32).

The method can further be characterized by the sparse directory structure (such as 402 depicted in FIG. 4) being formed by creating a first directory entry that includes a first address and a first forward link (such as INDEX 1 associated with FORWARD LINK 3

depicted in FIG. 4), and creating a second directory entry that includes a second address and a second forward link (such as INDEX 3 associated with FORWARD LINK 2). The method can further be characterized by determining that the second directory entry (such as VALUE 20) is located after the first directory entry (such as VALUE 7) in the data container (see pg. 9 lines 8-9) and thereby defining the first forward link to link to the second directory entry (such as links 312 depicted in FIG. 3). The method can further be characterized by creating a bottom level list (such as 306, 308 depicted in FIG. 3, see pg. 6 lines 29-30) that includes the first directory entry and the second directory entry. The method can further be characterized by creating a top level entry that comprises a link to the bottom level list, a lower range, and an upper range (such as 304 depicted in FIG. 3, see pg. 7 lines 8-9). The method can further be characterized by analyzing the bottom level list to determine the lower range and the upper range of the top level entry (see pg. 7 lines 9-11). The method can further be characterized by creating a top level directory that comprises the top level entry (such as 302 depicted in FIG. 3, see pg. 6 lines 29-30).

The method can further be characterized by creating a third directory entry associated with a third address and a third forward link, wherein the third address is between the first directory entry and the second directory entry (such as INDEX 4 having VALUE 15 depicted in FIG. 4). The method can further be characterized by determining that the third directory entry is located between the first directory entry and the second directory entry, and thereby changing the first forward link to link to the third directory entry (depicted in FIG. 4 as the FORWARD LINK associated with INDEX 1 being changed from 3 to 4).

The method can further be characterized by the sparse directory structure (such as 402 depicted in FIG. 4) including a first backward link (such as index 3 associated with

BACKWARD LINK 1) and a second backward link (such as index 2 associated with BACKWARD LINK 3). The method can further be characterized by determining that the first directory entry (such as VALUE 20) is located before the second directory entry (such as VALUE 37) in the data container and thereby defining the second backward link to link to the first directory entry (such as links 312 depicted in FIG. 3).

Claim 16 features a data storage system (such as 100 depicted in FIG. 1) that includes a data storage container (such as 104) and a controller (such as 102, see pg. 12-14). The controller is configured to define a sparse directory structure for the data container (see pg. 4 lines 21-22, pg. 5 lines 25-26), to determine that the sparse directory structure is to be changed (see pg. 6 lines 1-2), and to reconstruct the sparse directory structure into a fully populated directory structure (see pg. 4 lines 29-31, pg. 5 lines 8-10).

The sparse directory structure (such as 300 depicted in FIG. 3) can include a plurality of first directory entries (such as 310 in FIG. 3, see pg. 7 lines 3-6). The first directory entries can include an address to one of respective addressable spaces in the storage container (such as "VALUE" in FIG. 4, see pg. 7 lines 29-31). The first directory entries can also include a descriptor (such as "INDEX" in FIG. 4, see pg. 7 lines 27-28). The first directory entries can also include a link that points to a different one of the first directory entries (such as "FORWARD LINK" and "BACKWARD LINK" in FIG. 4, see pg. 7 lines 24-25, pg. 7 line 32 to pg. 8 line 5).

The sparse directory structure can be further characterized by a bottom level list of a plurality of the first directory entries (such as 306, 308 depicted in FIG. 3, see pg. 6 lines 29-30). The sparse directory can be further characterized by a top level entry for each of the bottom level lists (such as 304 depicted in FIG. 3, see pg. 6 lines 29-30, pg. 7 lines 8-9). The

sparse directory structure can be further characterized by a top level list that includes a plurality of the top level entries (such as 302 depicted in FIG. 3, see pg. 6 lines 29-30). In some embodiments the top level list can be a skip list, a linked list, or an ordered array (see pg. 7 lines 12-16). In some embodiments the bottom level lists can be a skip list or a linked list (see pg. 6 lines 31-32).

Claim 29 features a data storage system (such as 200 depicted in FIG. 2) having a controller (such as 102, see pg. 12-14) configured for selectively constructing either a variable size sparse directory structure for a data container (such as 300 depicted in FIG. 3, block 206 depicted in FIG. 2, see pg. 5 lines 4-6) or a fixed size fully populated directory structure for the same data container (such as 500 depicted in FIG. 5, block 216 depicted in FIG. 2, see pg. 5 lines 11-13). The fully populated directory structure has an entry for each addressable memory location in the data container (see pg. 4 lines 14-15). The sparse directory structure has fewer entries than a number of the addressable memory locations in the data container (see pg. 4 lines 11-13). The controller is configured for selectively reconstructing a previously constructed sparse directory structure into a fully populated directory structure (such as in block 212 of FIG. 2, see pg. 5 lines 8-10). The controller is also configured for selectively reconstructing a previously constructed fully populated directory structure into a sparse directory structure (see pg. 4 line 31 to pg. 5 line 2, pg. 6 lines 21-27).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-2, 16, and 29-33 stand rejected under 35 USC 102 as being anticipated by U.S. 2004/0059876 by Nanda ("Nanda '876").

Claims 3-5, 7-9, 17-19, 21-23, and 25-28 stand rejected under 35 USC 103 as being unpatentable over Nanda '876 in view of U.S. 2004/0221120 by Abrashkevich ("Abrashkevich '120").

VII. ARGUMENT

THE OFFICE HAS FAILED TO SUBSTANTIATE ANTICIPATION OF INDEPENDENT CLAIMS 1, 16, AND 29 BY NOT SHOWING THAT NANDA '876 DISCLOSES ALL THE RECITED FEATURES OF THOSE CLAIMS

Independent claims 1, 16, and 20 recite the following with respect to the *fully populated directory*:

> *determining that a sparse directory structure is to be changed; and*
> *reconstructing said sparse directory structure into a fully populated directory structure.*[2]
>
> *a controller that...determines that said sparse directory structure is to be changed, and reconstructs said sparse directory structure into a fully populated directory structure.*[3]
>
> *a controller configured for selectively constructing either a variable size sparse directory for a data container or a fixed size fully populated directory structure for the same data container.*[4]

Nanda '876 discloses two embodiments of a real time emulator of coherence directories in a shared memory multiprocessor system. The emulator disclosed in FIG. 1 and the descriptions thereof has a plurality of sparse directories 300, one connected to each of its memory nodes 100:[5]

---

[2] Excerpt of claim 1, emphasis added.
[3] Excerpt of claim 16, emphasis added.
[4] Excerpt of claim 29, emphasis added.
[5] See, for example, Nanda '876 para. [0019] lines 1-6.

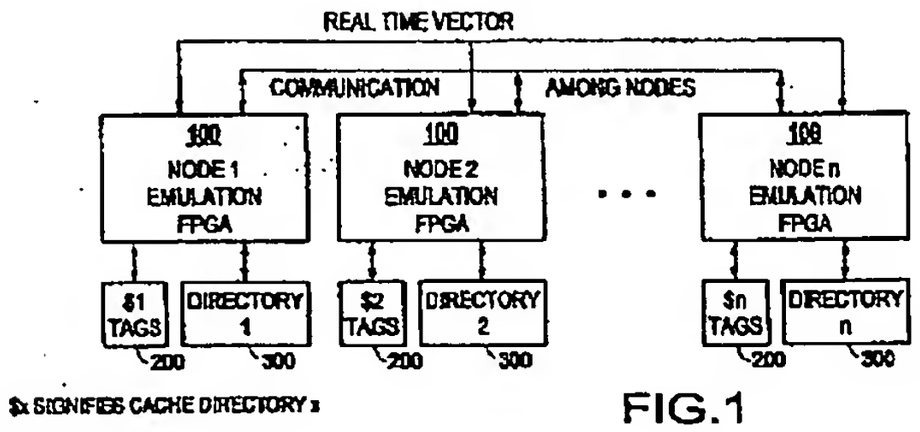

FIG.1

The emulator disclosed in FIG.4 and the descriptions thereof has a plurality of sparse directories 400, one connected to each of its memory nodes 100.[6]

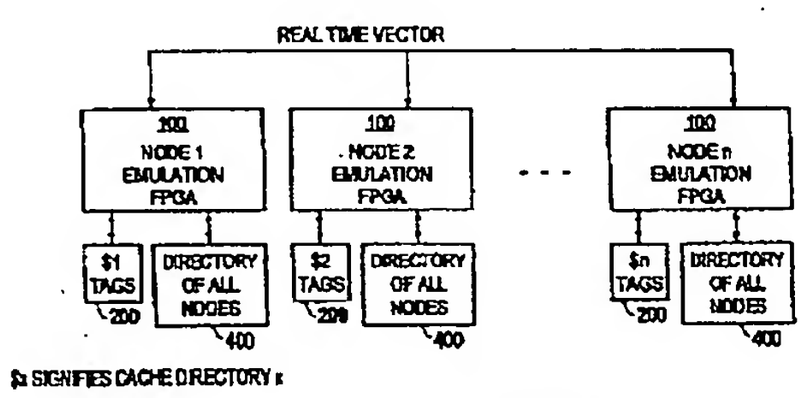

FIG.4

Nanda '876 explicitly defines a "sparse directory" as being limited to storing information associated with less than the entire addressable space of the memory node to which it is attached:

> One of the area efficient implementations of coherence directories is the "sparse directory" which has the capability to store the state of a limited number of memory lines. The sparse directory includes state information entries for only a subset of the memory lines of main memory.[7]

Because the sparse directories 300, 400 are limited in size, they must evict an existing entry to make room for a new entry when they become full.[8] The difference between the sparse

---

[6] See, for example, Nanda '876 para. [[0027] lines 1-4.
[7] Nanda '876 para. [0005] lines 12-16.
[8] Nanda '876 para. [0019] lines 16-18, para. [0032] lines 10-11.

directory 300 and the sparse directory 400 in Nanda '876 lies in what memory node(s) is/are associated with the information stored therein. The sparse directory 300 maintains information only on some memory lines of the particular node 100 to which it is connected.[9] Contrarily, the sparse directory 400, referred to as a "global sparse directory," maintains information on some memory lines of the particular node 100 to which it is connected as well as copies of the sparse directory entries for all the other nodes 100.[10]

Claims 1, 16, and 29 feature two different directory structures, the *sparse directory structure* and the *fully populated directory structure*. Claims 1 and 16 also feature *reconstructing* a directory structure from one of these two featured directory structures to the other structure. Claim 29 features *selectively constructing* either of these two featured directory structures for the same memory space. The skilled artisan, having read the specification, understands that the featured *sparse directory structure*[11] is constructed differently than the featured *fully populated directory structure*.[12] That is, unlike the *sparse directory structure*, the featured *fully populated directory structure* contains an entry for every single data block of the data space with which it is associated.[13]

However, the Office's rationale for the rejection of these claims is that Nanda '876 discloses evicting an entry from a sparse directory when it is full to make room for a new entry, and that allegedly anticipates the featured reconstructing the *sparse directory structure* into the *fully populated directory structure*.[14]

---

[9] Nanda '876 para. [0019] lines 4-6.
[10] Nanda '876 para. [0027] lines 3-7.
[11] Depicted in FIG. 3 of the specification.
[12] Depicted in FIG. 5 of the specification.
[13] See, for example, specification pg. 4 lines 14-20.
[14] Office Action of 3/26/2007 pg. 3 citing Nanda '876 para. [0032] lines 11-12.

During examination claims are given their "broadest reasonable interpretation consistent with the specification."[15] The "broadest reasonable interpretation" is the meaning that the skilled artisan would give to the claim term in view of the associated usage provided in the specification.[16] A construction that is inconsistent with the written description would not be arrived at by the skilled artisan, and is therefore not a "reasonable interpretation."[17]

The Office's claim construction requires that the meaning of both featured terms *sparse directory structure* and *fully populated directory structure* have the same structure. That is, the skilled artisan readily understands that when the sparse directory 300, 400 in Nanda '876 evicts an entry, its structure remains intact; the structure of a sparse directory. It is incontrovertible that such a construction is inconsistent with the usage of the term *fully populated directory structure* in the specification, where from a plain reading it is clear that the *fully populated directory structure* is constructed differently than the *sparse directory structure*. Again, the featured *sparse directory structure* has an entry for only a subset of the addressable spaces in the memory, whereas the featured *fully populated directory structure* has an entry for every addressable space in the memory.[18] It is also incontrovertible that such a construction is contrary to the featured *reconstructing said sparse directory* in claim 1,[19] which plainly means changing the structure of the sparse directory structure.

Therefore, the evidence in the record is conclusive that the skilled artisan would find that the Office's interpretation of at least the featured *fully populated directory structure* is inconsistent with both the plain meaning of the phrase in the context of the claim language,

---

[15] *Phillips v. AWH Corp.*, 75 USPQ2d 1321 (Fed. Cir. 2005)(en Banc); MPEP 2111

[16] *In re American Academy of Science Technical Center*, 70 USPQ2d 1827 (Fed. Cir. 2004); *In re Cortright*, 49 USPQ2d 1463, 1468 (Fed. Cir. 1999); *In re Morris*, 44 USPQ2d 1023 (Fed. Cir. 1997)

[17] *Phillips, supra*; *In re Morris, supra*; *In re Zletz*, 13 USPQ2d 1320, 1322 (Fed. Cir. 1989)

[18] Note 13.

and inconsistent with the usage of the phrase in the specification. The Office has failed to substantiate any evidence as to why the skilled artisan would agree that the sparse directory 300, 400 of Nanda '876 is within the broadest reasonable construction of the featured *fully populated directory structure*. As such, the Office's construction is not within the broadest reasonable interpretation consistent with the specification.

To substantiate anticipation in terms of Section 102, every element of the claimed invention must be identically shown in a single reference and arranged as in the rejected claim.[20] For the reasons set forth above, the Office has not substantiated evidence that Nanda '876 identically discloses the featured two different directory structures, the *sparse directory structure* and the *fully populated directory structure*, and the featured reconstructing one of the directory structures to the other of the directory structures. As such, the evidence in the record cannot sustain the present anticipatory rejection. The rejection is therefore reversible error, and as such Appellant respectfully requests that the rejection of claims 1, 16, and 29 and the claims depending therefrom be reversed.

THE OFFICE HAS FAILED TO SUBSTANTIATE OBVIOUSNESS OF CLAIMS 3-5, 7-9, 17-19, 21-23, AND 25-28

1. The Cited References Do Not Include All The Recited Features Of These Claims

A *prima facie* case of obviousness requires a showing of a teaching or suggestion for each claim limitation appearing in the claim.[21] In construing a claim term, the Office is obligated to apply to broadest reasonable interpretation consistent with the specification.[22]

---

[19] And similarly *reconstructs said sparse directory* featured in claim 16, and *selectively constructing either a...sparse directory structure...or a...fully populated directory structure* featured in claim 29.
[20] *In re Bond*, 15 USPQ2d 1566 (Fed. Cir. 1990).
[21] *In re Royka*, 180 USPQ 580 (CCPA 1974); MPEP 2143.
[22] *Phillips v. AWH Corp.*, 75 USPQ2d 1321 (Fed. Cir. 2005)(en banc); MPEP 2111.

An interpretation that is inconsistent with the specification is not reasonable.[23] Further, the Office is obligated to construe the claim language in accordance with its plain meaning.[24] Certainly, claim language cannot be reasonably be interpreted so broadly as to include meaning that is the opposite of what the ordinary meaning is where the ordinary meaning is consistent with the term's usage in the specification.[25]

For the reasons set forth above, the Office has not substantiated evidence that Nanda '876 includes the two different directory structures and the reconstructing of a directory structure featured by independent claims 1 and 16 from which these claims depend. In fact, the evidence shows that the Office's rationale requires a meaning that is opposite to the ordinary meaning consistent with usage in the specification for the featured *sparse directory structure* and *fully populated directory structure*. Appellant has shown in the record that Abrashkevich '120 fails to cure the deficiency of Nanda '876 in this regard, in that it discloses only one structure for its heap header.[26]

These rejected claims depend from one of independent claim 1 or 16, and thereby feature the differently structured *sparse directory structure* and *fully populated directory structure*. Therefore, there is lacking in the record the requisite evidence that the cited references include all the recited features of these rejected claims to substantiate obviousness.[27] The rejection is therefore reversible error, and as such Appellant respectfully requests reversal of the rejection.

---

[23] *In re Morris*, 44 USPQ2d 1023 (Fed. Cir. 1997).
[24] *Phillips, supra*; MPEP 2111.01.
[25] *In re Buszard*, 504 F.3d 1364 (Fed. Cir. 2007).
[26] See Appellant's Response of 4/3/2006 pg. 10.
[27] Examination Guidelines for Determining Obviousness in View of the Supreme Court Decision in *KSR International Co. v. Teleflex Inc.*, FR 57526

THE SKILLED ARTISAN WOULD NOT FIND IT DESIRABLE TO COMBINE/MODIFY THE CITED REFERENCES TO ARRIVE AT THE CLAIMED SUBJECT MATTER

An obviousness rejection must be evaluated in view of the *Graham*[28] factors: (a) the claimed invention must be considered as a whole; (b) the references must be considered as a whole and must suggest the desirability and thus the obviousness of making the combination; (c) the references must be viewed without the benefit of impermissible hindsight vision afforded by the claimed invention; and (d) reasonable expectation of success is the standard with which obviousness is determined.[29]

The subject matter of the rejected claims is adaptively mapping by employing one of two different directory structures, the *sparse directory structure* and the *fully populated directory structure*. The subject matter of Nanda '876 is directed to using sparse directories as part of a cache coherence emulator. The subject matter of Abrashkevich '120 is using a data structure employing a heap header to debug information stored in memory. The Office's entire basis for the motivation to combine the cited references to arrive at the subject matter of all these rejected dependent claims is the following:

> This provides an ability to reliably detect various types of memory errors, dynamically enable or disable memory debugging, enhance success of read and write operations using various memory verification techniques.[30]

A mere conclusory statement such as this is insufficient to establish *prima facie* obviousness. Rather, there must be some articulated reasoning with some rational

---

[28] *Graham v. John Deere*, 383 US 1 (1966).
[29] See MPEP 2141.

underpinning to support the legal conclusion of obviousness.[31] The Office's basis is conclusory because it neither provides a rationale for why the skilled artisan would reasonably view either of the cited references as teaching or suggesting the featured use of two differently constructed directory structures, and the featured reconstructing a directory structure from one structure to the other structure.

There is lacking in the record substantiating evidence for obviousness that the skilled artisan, having knowledge of the cited references, would find it obvious to modify and combine them to arrive at the present embodiments as claimed.[32] The rejection is therefore reversible error, and as such Appellant respectfully requests reversal of the rejection.

Conclusion

In conclusion, Appellant respectfully requests that the rejection of all pending claims be reversed.

Respectfully submitted,

By: _____

Mitchell K. McCarthy, Reg. No. 38,794
McCarthy Law Group
3580 Northwest Expressway, #353
Oklahoma City, Oklahoma 73102
www.mccarthyiplaw.com
Telephone: (405) 601-2798
Customer No. 71222

---

[30] Office Action of 3/26/2007 pg. 5, citing Abrashkevich Abstract lines 3-7.
[31] *In re Kahn*, 441 F.3d 997, 988 (Fed. Cir. 2006); *KSR v. Teleflex*, 127 US 1727 (2007).
[3232] *Supra* note 9.

VIII. CLAIMS APPENDIX

1. (Original) A method of maintaining a directory for a data container comprising:

determining that a sparse directory structure is to be changed; and

reconstructing said sparse directory structure into a fully populated directory structure.

2. (Original) The method of claim 1 further comprising:

determining that said fully populated directory structure is to be changed; and

reconstructing said fully populated directory structure into a sparsely populated directory structure.

3. (Original) The method of claim 1 wherein said sparse directory structure comprises:

a plurality of first directory entries comprising an address to one of said addressable spaces, a descriptor, and at least one link, said link being a pointer to a different of said directory entries;

at least one bottom level list comprising at least one of said plurality of first directory entries;

at least one top level entry for each of said bottom level lists; and

a top level list comprising said top level entries.

4. (Original) The method of claim 3 wherein said top level list is a skip list.

5. (Original) The method of claim 3 wherein said top level list is a linked list.

6. (Canceled)

7. (Original) The method of claim 3 wherein said top level list is an ordered array.

8. (Original) The method of claim 3 wherein said bottom level lists are skip lists.

9. (Original) The method of claim 3 wherein said bottom level lists are linked lists.

Claims 10-15 (Cancelled).

16. (Original) A data storage system comprising:

a data storage container; and

a controller that defines a sparse directory structure for said data container,

determines that said sparse directory structure is to be changed, and

reconstructs said sparse directory structure into a fully populated directory structure.

17. (Original) The data storage system of claim 16 wherein said sparse directory structure comprises:

a plurality of first directory entries comprising an address to one of said addressable spaces, a descriptor, and at least one link, said link being a pointer to a different of said directory entries;

at least one bottom level list comprising at least one of said plurality of first directory entries;

at least one top level entry for each of said bottom level lists; and

a top level list comprising said top level entries.

18. (Original) The data storage system of claim 17 wherein said bottom level list is a skip list.

19. (Original) The data storage system of claim 17 wherein said bottom level list is a linked list.

20. (Canceled)

21. (Original) The data storage system of claim 17 wherein said bottom level list is an ordered array.

22. (Original) The data storage system of claim 17 wherein said top level list is a skip list.

23. (Original) The data storage system of claim 17 wherein said top level list is a linked list.

24. (Canceled)

25. (Original) The data storage system of claim 17 wherein said top level list is an ordered array.

26. (Previously presented) The method of claim 1, wherein the sparse directory structure of the determining step is formed by steps comprising:

creating a first directory entry comprising a first address, and a first forward link;

creating a second directory entry comprising a second address, and a second forward link;

determining that said second directory entry is located after said first directory entry in said data container;

defining said first forward link to link to said second directory entry;

creating a bottom level list that comprises said first directory entry and said second directory entry;

creating a top level entry that comprises a link to said bottom level list, a lower range, and an upper range;

analyzing said bottom level list to determine said lower range and said upper range of said top level entry; and

creating a top level directory that comprises said top level entry.

27. (Previously presented) The method of claim 26 wherein said first directory entry comprises a first backward link and said second directory comprises a second backward link, the method further comprising:

determining that said first directory entry is located before said second directory entry in said data container; and

defining said second backward link to link to said first directory entry.

28. (Previously presented) The method of claim 26 further comprising:

creating a third directory entry comprising a third address, and a third forward link, said third address being between said first directory entry and said second directory entry; and

adding said third directory entry by steps comprising:

adding said third directory entry to said bottom level list;

determining that said third directory entry is located between said first directory entry and said second directory entry; and

changing said first forward link to link to said third directory entry; and

defining said third forward link to link to said second directory entry.

29. (Previously presented) A data storage system comprising a controller configured for selectively constructing either a variable size sparse directory structure for a data container or a fixed size fully populated directory structure for the same data container.

30. (Previously presented) The data storage system of claim 29 wherein the fully populated directory structure comprises an entry for each addressable memory location in the data container.

31. (Previously presented) The data storage system of claim 30 wherein the sparse directory structure comprises fewer entries than a number of the addressable memory locations in the data container.

32. (Previously presented) The data storage system of claim 29 wherein the controller is configured for selectively reconstructing a previously constructed sparse directory structure into a fully populated directory structure.

33. (Previously presented) The data storage system of claim 29 wherein the controller is configured for selectively reconstructing a previously constructed fully populated directory structure into a sparse directory structure.

IX. EVIDENCE APPENDIX

No additional evidence is included.

X. RELATED PROCEEDINGS APPENDIX

There exist no relevant related proceedings concerning this Appeal before the Board.